Anonymizing datasets with demographics and diagnosis codes in the presence of utility constraints

Giorgos Poulis, Grigoris Loukides, Spiros Skiadopoulos, Aris Gkoulalas-Divanis

Abstract

Publishing data about patients that contain both demographics and diagnosis codes is essential to perform large-scale, low-cost medical studies. However, preserving the privacy and utility of such data is challenging, because it requires: (i) guarding against identity disclosure (re-identification) attacks based on both demographics and diagnosis codes, (ii) ensuring that the anonymized data remain useful in intended analysis tasks, and (iii) minimizing the information loss incurred by anonymization, to preserve the utility of general analysis tasks that are difficult to determine before data publishing. Existing anonymization approaches are not suitable for being used in this setting, because they cannot satisfy all three requirements. Therefore, in this work, we propose a new approach to deal with this problem. We enforce the requirement (i) by applying \((k,k^m)\)-anonymity, a privacy principle that prevents re-identification from attackers who know the demographics of a patient and up to \(m\) of their diagnosis codes, where \(k\) and \(m\) are tunable parameters. To capture the requirement (ii), we propose the concept of utility constraint for both demographics and diagnosis codes. Utility constraints limit the amount of generalization and are specified by data owners (e.g., the healthcare institution that performs anonymization). We also capture requirement (iii), by employing well-established information loss measures for demographics and for diagnosis codes. To realize our approach, we develop an algorithm that enforces \((k,k^m)\)-anonymity on a dataset containing both demographics and diagnosis codes, in a way that satisfies the specified utility constraints and with minimal information loss, according to the measures. Our experiments with a large dataset containing more than 200,000 electronic health records show the effectiveness and efficiency of our algorithm.

1. Introduction

Healthcare organizations collect increasingly large amounts of data, including clinical trials, Electronic Health Records (EHR), disease registries, and medical imaging databases. In fact, the estimated amount of healthcare data in the world was 0.5 Exabytes (0.5 \(\times\) \(10^{18}\) bytes) in 2012 and is expected to reach 25 Exabytes by 2020 [66]. Healthcare data are essential for performing large-scale, low-cost analyses [18], which range from Genome-Wide Association Studies (GWAS) to predictive modeling [9,32] and have the potential to improve medical research and practice. For instance, the study in [14] used more than 350,000 records of the Scandinavian Donations and Transfusions database, along with the donors’ and the recipients’ health records, to answer whether blood transfusions transmit cancer, and it had a substantial impact on public health policies regarding restrictions placed on blood donors [3,13]. Another study [75] used an EHR database of over 300,000 records, to learn meaningful comorbidities, which are associated with different stages of Chronic Obstructive Pulmonary Disease (COPD). This study has the potential to improve COPD prognosis, drug development, and clinical trial design.

While the value of analyzing healthcare data is widely recognized, data sharing remains an obstacle for the majority of healthcare providers [17]. In particular, the privacy-preserving sharing of healthcare data beyond authorized recipients (e.g., researchers or employees of the institution that has collected the data) is challenging [15,16,25]. This is partly because it cannot be facilitated based on access control and encryption-based methods [59,65], or by relying solely on policies (e.g., the HIPAA Privacy Rule [57] in the US, the Anonymization Code [56] in the UK, and the Data Protection Directive [58] in the EU). In fact, a major concern in
healthcare data publishing is identity disclosure (or reidentification), an attack in which patients are linked with their records in the published dataset. Identity disclosure can be performed, even when the published dataset is devoid of direct identifiers (e.g., patient phone numbers), due to the availability of external data sources that can be linked to the published dataset, based on demographics [67] or diagnosis codes [47]. For example, Sweeney estimated that 87% of US citizens can be re-identified based on gender, date of birth, and ZIP code, while Golle [27] estimated this percentage as 63%, using newer, using newer, using newer 2000 data. In addition, Loukides et al. [44] showed that 96% of 2700 patients, who are involved in an NIH-funded GWAS, are uniquely re-identifiable based on their set of diagnosis codes. In response, various methods have been proposed to prevent identity disclosure when publishing a dataset that contains demographic attributes (e.g., [16,39,64,79]), or diagnosis codes (e.g., [24,31,47,70]).

In this work, we consider the problem of preventing identity disclosure when we need to publish datasets containing both demographics and diagnosis codes, henceforth termed RT-datasets. Such datasets are used in many applications [61]. Here, we provide some recent examples:

1. The CMS-HCC risk adjustment model [28] uses demographics and diagnoses of health insurance beneficiaries, to predict the health costs of a US health insurance program, called Medicare Advantage. In particular, beneficiaries’ demographics (e.g., gender, aged/disabled status, and whether a beneficiary lives in a certain community or close to an institution) and diagnostic data are used to build and update the risk model. The data are provided from hospital inpatient, hospital outpatient, and physical risk adjustment data.

2. Various epidemiological [11,26,62] and cancer research [36] studies are based on data containing demographics and diagnosis codes of patients in New South Wales (NSW), Australia. For example, the study of [36] used the data of women over 45 who are associated with certain diagnosis and procedural codes indicating invasive breast cancer. These data were obtained from the NSW Cancer Registry and from several routinely-collected administrative and self-reported health datasets in NSW, and they were analyzed to find out their predictive power in identifying invasive breast cancer cases.

3. The study of [7] uses a dataset containing demographics and ICD-9 diagnosis codes of patients from various US hospitals, to identify groups of patients that are likely to be diagnosed with diabetes, based on their demographics. In particular, it uses multi-label learning algorithms [74], to estimate the risk that each patient has for being diagnosed with diabetes, based on multiple demographics, such as race, gender, and age group.

These applications use data that are devoid of direct identifiers and thus potentially susceptible to identity disclosure. However, their authors recognize the need for algorithms that anonymize both demographics and diagnosis codes, in order to prevent identity disclosure [7] and increase data availability [36]. Also, publishing RT-datasets is important to support analysis tasks, including case count studies [46,54], which require accurately counting the number of patients associated with specific demographics and diagnosis codes, predictive modeling, and query answering [61].

However, anonymizing an RT-dataset in a utility-preserving way is a very challenging task. This was acknowledged in [54], which is the first work that studied the general problem of anonymizing an RT-dataset. As explained in Section 2, our work differs from that of [54] in terms of five main dimensions (anonymization principle, data transformation operation, support of utility constraints, information loss criterion, and anonymization algorithm). Specifically, there are three challenges entailed in the anonymization of an RT-dataset in a utility-preserving way. First, identity disclosure cannot be prevented by applying existing methods on demographics and on diagnosis codes separately. This is because an attacker with knowledge of both demographics and diagnosis codes can still re-identify a patient, when the combination of demographics and diagnosis codes of the patient is unique in the anonymized dataset [61]. Specifically, the probability of re-identifying a patient based on such a combination is the reciprocal of the frequency of the combination in the anonymized dataset. Second, data utility must be preserved. This requires constructing an anonymized RT-dataset which allows performing: (i) intended analysis tasks with no loss of accuracy and (ii) general analysis tasks, which are difficult to determine before data publishing, with minimum loss of accuracy.

However, the existing methods for anonymizing RT-datasets [54,61,68] may substantially reduce the accuracy of intended analysis tasks, or incur excessive information loss, which reduces the accuracy of general analysis tasks. Specifically, the method of [54] does not preserve data truthfulness, because it is based on noise addition. That is, it produces synthetic data. Such data are useful for general statistical analysis or mining tasks and can offer strong privacy guarantees [54]. However, the fact that they contain fake information about patients makes them unsuitable for certain applications. For example, they may lead to false alarms in epidemiology [8]. Therefore, our focus is on an anonymization approach that produces truthful data. In addition, the methods of [61,68] do not preserve both aspects of data utility; the output of [61] is of little use in intended tasks and that of [68] incurs substantial information loss, which affects the output of general analysis tasks. To illustrate the challenges of anonymizing an RT-dataset, we provide Example 1.

Example 1. Consider the RT-dataset D in Fig. 1a. Age, Origin and Gender are demographic attributes, and Disease is a set containing diagnosis codes, whose description is presented in Fig. 2. The dataset in Fig. 1b was produced by applying the method of [78] on the demographic attributes and the method of [45] on Disease. In particular, the latter dataset satisfies 2-anonymity [67] and 2^2-anonymity [50,71], because no record contains a unique combination of demographic values, or a unique combination of two or fewer diagnosis codes (the result of generalizing diagnosis codes is enclosed in () and interpreted as any combination of the codes). However, an attacker who knows the demographics and two diagnosis codes of a patient can still re-identify patients. For example, an attacker who knows that Zoe is a 30-year-old female from Spain, diagnosed with 493.2 (Chronic obstructive asthma) and 494.1 (Bronchiectasis with acute exacerbation), can associate her with the record 3 of the dataset in Fig. 1b.

Consider also that the RT-dataset in Fig. 1a needs to support a study which requires counting the number of patients who are at most 50 and are diagnosed with 494.1. Applying the method of [61] (respectively, of [68]) produces the anonymized dataset in Fig. 1c (respectively, in Fig. 1d). However, these datasets cannot support the study, since the number of records corresponding to patients at most 50 who are diagnosed with 494.1 cannot be accurately determined. This is because the Age values of the records 0 to 4 have been replaced with the range (interval) [19:51] in Fig. 1c, while 494.1 has been generalized together with other diagnosis codes in Fig. 1d.

To address these challenges, our work makes the following specific contributions.

1 Utility constraints for RT-datasets. We investigate how to model and enforce the requirement of supporting case count studies with no accuracy loss. We propose the concept of utility constraint for RT-datasets, building upon previous work on
The utility constraint set, $\mathcal{UC}_k$, restricts the utility of the anonymized dataset to a specified level. The concept of utility constraint motivates the development of anonymization algorithms that meet these constraints to ensure the utility of the published dataset is maintained. For example, when $\mathcal{UC}_k$ is used, the anonymization process must ensure that the information loss is below a certain threshold, ensuring the record remains useful for the intended study. The anonymization algorithms, such as $\text{ART}_{\mathcal{UC}}$, are designed to achieve this by minimizing the information loss of the dataset in Fig. 1a. Notice that the probability of re-identification is reduced to $\frac{1}{k}$ when $k$ is large, where $k$ is a parameter selected by data owners (e.g., the data collecting healthcare institution) to protect patient privacy. 

**Algorithm 2:** Anonymization algorithm for $RT$-datasets

Given an $RT$-dataset $D$, the algorithm $\text{ART}_{\mathcal{UC}}$ processes the demographic attributes and diagnosis codes separately. Specifically, it partitions the dataset into small clusters with similar values in the demographic attributes and generalizes the values in each cluster. After that, $\text{ART}_{\mathcal{UC}}$ merges clusters with the same generalized values and anonymizes the diagnosis codes in each cluster. The anonymization employs generalization and suppression and enforces $(k, k^m)$-anonymity in a utility-preserving way. For example, $\text{ART}_{\mathcal{UC}}$ produced the $(2, 2^2)$-anonymous dataset in Fig. 3, when applied to the dataset in Fig. 1a. Notice that the probability of re-identification based on the demographics and up to 2 diagnosis codes is at most $\frac{1}{4}$, while the anonymized dataset supports the study of Example 1. In addition, the information loss is lower compared to the datasets produced by methods of [61, 68] (illustrated in Fig. 1c and d respectively).
Evaluation of ARTUC. We investigate the effectiveness and efficiency of ARTUC by conducting experiments using a publicly available RT-dataset having 5 demographic attributes and over 30,000 electronic health records and an RT-dataset having 2 demographic attributes and over 200,000 electronic health records. We also use 9 utility constraint sets, which simulate the requirement of supporting different type of case count studies. Our results show that our algorithm performs anonymization according to the utility constraints and with low information loss. In addition, it takes less than 10 min and scales well with the dataset size and anonymization parameters.

The rest of the paper is organized as follows. Section 2 surveys related work. Section 3 presents the fundamental concepts used in this work and the problem statement. Section 4 presents the ARTUC algorithm, and Section 5 the experimental evaluation. Section 6 discusses extensions and limitations of our approach. Last, Section 7 concludes the paper.

2. Related work

In this section, we discuss anonymization methods that are closer to ours (see [15,52,25] for surveys). For extensive surveys on anonymization principles and methods for healthcare data, the reader is referred to [17,25,27]. In particular, Section 2.1 discusses algorithms for anonymizing RT-datasets, and Section 2.2 provides a brief overview of algorithms for anonymizing demographics and diagnosis codes.

2.1. Anonymization methods for RT-datasets

As mentioned in Section 1, methods for anonymizing an RT-dataset have been proposed in [54,61,68]. More specifically, the method of Takahashi et al. [68] enforces \(k\)-anonymity [67], which requires at least \(k\) records to have the same values in all demographic attributes and in the set-valued attribute containing diagnosis codes. To enforce \(k\)-anonymity, it employs hierarchy-based generalization [20], effectively replacing each group of values with their closest common ancestor in a given hierarchy. Furthermore, it applies pre-generalization, which generalizes some values in a predetermined way, selected by data owners. Thus, the method of [68] treats both demographics and diagnosis codes in the same way. As a result, it explores a much smaller space of potential solutions than our algorithm and incurs higher information loss. This is because it: (i) pre-generalizes attribute values, even when they can be generalized with lower information loss (i.e., replaced with more specific values), and (ii) employs the hierarchy-based generalization model for diagnosis codes, which was shown to offer lower data utility than the set-based generalization model we adopt [46].

Poulis et al. [61] proposed a general framework for enforcing \((k, k^m)\)-anonymity on relational and set-valued attributes. In this framework, the \(R_{Me}\) algorithm stands out, since it anonymizes RT-datasets with low information loss [61]. When applied to an RT-dataset, \(R_{Me}\) generalizes demographic attributes so the incurred information loss remains lower than a given threshold and performs set-valued generalization to diagnosis codes. However, none of the algorithms in [61] allows performing intended case count studies with no loss of accuracy, as our algorithm, ARTUC, does. Furthermore, ARTUC outperforms \(R_{Me}\) in terms of preserving data utility, as shown in our experiments.

Another method for anonymizing an RT-dataset was proposed by Mohammed et al. [54]. The method enforces differential privacy [12], a strong privacy principle, which ensures that the presence or absence of information about an individual in the dataset does not significantly affect the outcome of analysis applied to the dataset. In other words, any inference that an attacker can make about an individual will be (approximately) independent of whether the individual’s record is contained in the dataset or not. To enforce differential privacy, the method of [54] constructs a generalized contingency table, which records the counts of all combinations of values, and then adds noise to the counts to satisfy differential privacy. The utility goal of the method is to preserve as much information as possible for building a classifier.

Consequently, the method of [54] differs from our work along five main dimensions:

1. It employs differential privacy, instead of \((k, k^m)\)-anonymity. Thus, it aims to protect the inference of any information about an individual. On the other hand, our method aims to protect from identity disclosure (and can be extended to additionally protect from attribute disclosure, as explained in Section 6).

2. It produces a synthetic dataset, which is the result from adding noise to counts, in order to enforce privacy. Thus, it considers applications where data truthfulness is not necessary. On the other hand, our work employs generalization and/or suppression, which guarantee data truthfulness and make the result of our method suitable for more applications.

3. It does not consider utility constraints, unlike our work. Thus, it assumes a setting where any change to the values of demographics and/or diagnosis codes is acceptable. By employing utility constraints, our work is applicable to a more general setting.

4. It considers a different information loss criterion than our work. That is, it aims to avoid information loss that harms the task of classification. As acknowledged by the authors of [55], their approach is not developed for “general analysis tasks that focus on attribute values of individual records”. On the other hand, our approach adopts general utility measures for demographics and for diagnosis codes, which do not consider a specific task and are able to quantify the uncertainty in interpreting generalized values [46,79].

5. It employs a fundamentally different algorithm to split the dataset into groups. Specifically, the algorithm employed in [54] works in a top-down fashion. It starts from the coarsest possible partition of the dataset (i.e., all records are contained in the same group) and then iteratively partitions the dataset (i.e., splitting the group into smaller groups), along a single
attribute at a time. On the other hand, our algorithm works by first creating groups of records around utility constraints based on demographics and then creating clusters in a bottom-up fashion. Bottom-up anonymization algorithms allow for grouping data generally exploring a larger space of potential solutions that often leads to preserving data utility better [79].

2.2. Anonymization methods for demographics and for diagnosis codes

A multitude of algorithms can be used to anonymize demographics. These algorithms can be categorized based on two dimensions: (i) their privacy principle, and (ii) data transformation strategy, as shown in Table 1. Observe that these algorithms employ the principles of k-anonymity [67], l-diversity [51], and τ-closeness [42], which protect from different attacks, and transform data using generalization and/or suppression [67], microaggregation [10], or bucketization [76]. In the following, we discuss algorithms that apply k-anonymity using generalization and/or suppression, since they are closer to ours. However, these algorithms are not alternatives to our method, because they cannot prevent identity disclosure in RT-datasets (i.e., protect only the demographics). These algorithms can be classified into three categories, based on the way they work: (i) Lattice-based, (ii) Partitioning-based, and (iii) Clustering-based.

Lattice-based algorithms employ a lattice to encode all the ways of generalizing demographics, and they search the lattice for ways that satisfy k-anonymity with minimum information loss. The search can be performed based on binary-search [63], Apriori-like heuristics [2], and genetic algorithms [34]. Lattice-based algorithms typically incur more information loss than Partitioning [33,40] and Clustering-based algorithms [4,23,41,78], which work by creating groups of records albeit in different ways. For example, the Partitioning-based algorithm in [40] creates groups using the kd-tree construction mechanism [19], while the Clustering-based algorithm in [78] creates clusters as in bottom-up hierarchical clustering [29]. Generally, Clustering-based algorithms incur a lower amount of information loss compared to the Partitioning-based ones. Our algorithm, ArtUC, performs the anonymization of demographics, as the Clustering-based algorithms do.

There are also various algorithms for anonymizing diagnosis codes. These algorithms can be categorized according to their privacy principle and data transformation strategy, as can be seen in Table 2. These algorithms employ k'-anonymity [71], privacy-constrained anonymity [46], (h,k,p)-coherence [80], p-uncertainty [5], and PS-rule based anonymity [49], and they transform diagnosis codes using generalization and/or suppression. In the following, we discuss algorithms for enforcing k'-anonymity. These algorithms are closer to ours, because they aim to prevent identity disclosure. However, they are not alternatives to our method, because they cannot protect RT-datasets (i.e., protect only the diagnosis codes). Terrovitis et al. [71] proposed an algorithm, called Apriori Anonymization (AA), which works iteratively in a bottom-up fashion. In each iteration, it identifies all sets of diagnosis codes of a certain size i (initially i = 1) and applies generalization to make the dataset k'-anonymous. Then, it considers all sets of diagnosis codes of size i + 1 and repeats the same process. When i = m the process ends and a k'-anonymous dataset is produced. However, AA does not allow performing intended case count studies with no loss of accuracy [46]. In response, Loukides et al. [50], proposed a method for enforcing k'-anonymity by disassociation [70], an operation that splits the diagnosis codes in records into non-overlapping subsets. The method first creates clusters of records that have semantically similar diagnosis codes and then applies disassociation to the records of each cluster to enforce k'-anonymity. However, the method of [50] does not preserve data truthfulness. Loukides et al. also proposed the UGACLIP and CBA algorithms in [46,45], respectively. These algorithms apply generalization and suppression, and they aim to prevent re-identification based on specific sets of diagnosis codes that are provided as input. They apply generalization and suppression in a way that preserves specific associations between diagnosis codes, which are modeled as utility constraints. CBA is more effective than UGACLIP in terms of preserving the specified associations and in terms of incurring low information loss. In ArtUC, we use an adaptation of CBA to anonymize diagnosis codes.

3. Background and problem statement

This section discusses the fundamental concepts that are used in our approach, and it provides the problem statement.

3.1. RT-datasets and their protection against identity disclosure

We consider a dataset in which every record corresponds to a distinct patient and contains a number of demographic attributes, as well as a set-valued attribute containing diagnosis codes. Such a dataset is referred to as an RT-dataset. Without loss of generality, we assume that the first l attributes in an RT-dataset correspond to the demographic attributes, A1,..., Al, and the last attribute, A1+l, is a set containing the diagnosis codes of the patient. Extensions to this modeling dealing with multiple set-valued attributes are discussed in Section 6. Each demographic attribute takes values from a different domain, which can be numerical (e.g., for Age) or categorical (e.g., for Gender). Following [67], we also assume that there
is an underlying hierarchy, for each categorical demographic attribute. On the other hand, the attribute $A_{k+1}$ is a subset of the set of ICD-9 codes [46]. In addition, following existing works [31,44,46,50], we consider ICD-9 codes. If some records contain ICD-10 codes, they can be easily mapped to ICD-9 codes, using General Equivalence Mappings, as explained in [6].

Let us now explain how identity disclosure can be performed by an attacker on an RT-dataset. We assume that the attacker knows all the values of a patient in each demographic attribute, $A_1,\ldots, A_k$, and up to $m$ diagnosis codes of the attribute $A_{k+1}$. Knowledge about demographic attributes can be obtained from linking external datasets (e.g., voter lists with hospital discharge summaries), as explained in [67]. In addition, knowledge about diagnosis codes may be solicited from external data sources, including the electronic health record system, as explained in [46], or be background knowledge. The parameter $m$ is an integer, which is set by data owners. The values of $m$ range from 0 to the number of distinct diagnosis codes contained in the dataset. The minimum value does not prevent identity disclosure based on diagnosis codes, while the maximum value protects from attackers with knowledge about any combination of diagnosis codes that a patient may have. Commonly, data owners set $m$ to a small constant [50], since it is unlikely for attackers to know many diagnosis codes for a certain individual.

To prevent identity disclosure based on the aforementioned knowledge, we employ $(k,k^m)$-anonymity [61], which is defined below.

**Definition 1.** An RT-dataset is $(k,k^m)$-anonymous, when an attacker who knows:

1. any combination of the demographic attributes of a patient, and
2. any combination of at most $m$ diagnosis codes of the patient, cannot use this knowledge to distinguish a record from at least other $k-1$ records in the dataset, where $k$ and $m$ are anonymization parameters specified by data owners.

$(k,k^m)$-anonymity limits the probability of identity disclosure, based on the knowledge specified in Items 1 and 2 of **Definition 1**, to at most $1/k$. Clearly, larger values of $k$ and $m$ achieve higher privacy protection. For example, the dataset in Fig. 3 is $(2,2^2)$-anonymous, because an attacker who knows the values of a patient in the set of demographic attributes (Age, Origin, Gender), as well as up to 2 diagnosis codes, cannot re-identify the patient with probability larger than $\frac{1}{4}$.

Observe that $(k,k^m)$-anonymity provides the same protection as $k$-anonymity [64], for demographic attributes, and as $k^m$-anonymity [72], for the diagnosis codes attribute. However, the inverse does not hold. Specifically, an RT-dataset may be $k$ and $k^m$ but not $(k,k^m)$-anonymous. For instance, the dataset of Fig. 1b is both 2-anonymous and 2$^2$-anonymous, but it is not $(2,2^2)$-anonymous. Therefore, an attacker who knows the demographics and up to 2 diagnosis codes of a patient may re-identify the patient, as discussed in **Example 1**. On the contrary, this is not possible when the $(2,2^2)$-anonymous dataset of Fig. 3 is used instead, because at least two records could belong to the patient.

1 A hierarchy is a tree structure whose leaves represent the original values in an attribute and internal nodes represent more abstract values that summarize their descendants in the tree. See, for example, Fig. 4.

**Fig. 5.** Example of applying generalization to the cluster containing the records (6,7) of the dataset of Fig. 1a.

### 3.2. Generalization, suppression and data utility quantification

To enforce $(k,k^m)$-anonymity, we may employ generalization and/or suppression. Generalization replaces a value with a more general value, while suppression deletes a value. Both operations have been applied to anonymize demographics and/or diagnosis codes (see Section 2). Suppression is a more drastic operation than generalization, but it is useful to avoid generalizations of diagnosis codes that have very low data utility [46]. For generalization, we apply a local recoding [39] generalization model that is inspired by [61]. Given a cluster of records $C$, our model replaces the values in each attribute of the records in the cluster, as explained below.

**Definition 2.** Given a cluster $C$, generalization replaces the values of all records in $C$, in each attribute $A$, as follows:

- **If $A$ is a numerical demographic attribute,** the values are replaced by the range (interval) comprised of the minimum and maximum of these values.
- **If $A$ is a categorical demographic attribute,** the values are replaced by their closest common ancestor in the hierarchy of $A$. The common closest ancestor is the node that belongs to every path from a value to the root of the hierarchy and is as far as possible from the root.
- **If $A$ is the set-valued attribute,** each value (set of diagnosis codes) of a record in $C$ is replaced by one or more sets of diagnosis codes.

Thus, the generalization model replaces the value of a record in the cluster, with respect to an attribute, with a node in the hierarchy of the attribute, or with a range, or with set(s) of diagnosis codes, depending on the type of the attribute. For brevity, we refer to the result of applying generalization to a value in a demographic attribute (i.e., a node in the hierarchy, or range) as generalized value, and to a set of diagnosis codes that is created by applying generalization to the value in the set-valued attribute as generalized diagnosis code. A generalized value is interpreted as a single node in the hierarchy or a range, whereas a generalized diagnosis code is interpreted as any subset of the diagnosis codes that it has replaced.

Note that there is a single way to replace the values in a cluster with respect to a demographic attribute, whereas there are multiple ways to perform the same for the diagnosis codes in the set-valued attribute. That is, our generalization model does not specify exactly which generalized diagnosis codes will be created. This allows our anonymization algorithm to select the best way to create generalized diagnosis codes, among a large number of possible alternatives (exponential in the number of distinct codes in the cluster), which is important to preserve data utility. **Example 2** below illustrates our generalization model.

**Example 2.** Consider a cluster containing the records 6 and 7 of the RT-dataset in Fig. 1a and the result of applying the generalization model to the cluster, shown in Fig. 5. Observe that the values [44,47] in the numerical demographic attribute Age have been replaced by the range [44,47]. This implies that the records in the cluster may have any value from 44 to 47 in Age. In addition, the values [Ghana, Portugal] in the categorical demographic attribute Origin have been replaced by their closest common ancestor All in the hierarchy of the attribute.  

<table>
<thead>
<tr>
<th>id</th>
<th>Age</th>
<th>Demographics</th>
<th>Gender</th>
<th>Diagnosis codes</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>44-47</td>
<td>All</td>
<td>All</td>
<td>494.1 (458.1, 458.21)</td>
<td>494.1</td>
</tr>
<tr>
<td>7</td>
<td>44-47</td>
<td>All</td>
<td>All</td>
<td>(458.1, 458.21)</td>
<td>494.1</td>
</tr>
</tbody>
</table>
the hierarchy for Origin, which is shown in Fig. 4. Similarly, the values {Male, Female} in the categorical demographic attribute Gender have been replaced by their closest common ancestor All in the hierarchy for Gender of Fig. 4. Moreover, the diagnosis codes have been replaced by the generalized diagnosis codes (494.1), and (458.1, 458.21). The generalized diagnosis code (494.1) is interpreted as Chronic hypotension, and the generalized diagnosis code (458.1, 458.21) is interpreted as Chronic hypotension and/or Hypotension of hemodialysis.

Clearly, our generalization model can enforce \((k, k^m)\)-anonymity on an RT-dataset, when the following conditions hold:

**Condition 1.** Each cluster contains at least \(k\) records with the same generalized value, in each demographic attribute.

**Condition 2.** Each combination of \(m\) or fewer diagnosis codes appears in at least \(k\) records of the cluster (a diagnosis code appears in a generalized diagnosis codes when it has been replaced by it after generalization).

Condition 1 (respectively, Condition 2) ensures that the attacker cannot use their knowledge about the demographics (respectively, the diagnosis codes) of the patient in Definition 1 to re-identify the patient with probability that exceeds \(1/k\).

There are many different ways to anonymize an RT-dataset, using generalization and/or suppression, which do not offer the same utility. Typically, the utility of an anonymized dataset in general analysis tasks is measured based on: (a) the amount of information loss incurred by anonymization and (b) the accuracy of answering aggregate queries using the anonymized dataset.

To measure the amount of information loss, there are various utility measures that are applicable to demographics [4,22,39,79] or diagnosis codes [30,47,72]. For demographics, we use the NCP (Normalized Certainty Penalty) measure [79], due to its flexibility in dealing with both numerical and categorical demographic attributes and its ability to accurately quantify the uncertainty in interpreting generalized values [79]. The following definitions explain the NCP measure.

**Definition 3.** Given a generalized value \(\bar{u}\) in a demographic attribute \(A\), the NCP of \(\bar{u}\) is defined as:

\[
\text{NCP}_A(\bar{u}) = \begin{cases} 
0, & |\bar{u}| = 1 \\
|\bar{u}|/|A|, & \text{otherwise} 
\end{cases}
\]

where \(\bar{u}\) and \(|A|\) are defined as follows.

If \(A\) is numerical, \(\bar{u}\) is the length of the range \(\bar{u}\) and \(|A|\) is the domain size of \(A\).

If \(A\) is categorical, \(\bar{u}\) is the number of leaves of the subtree rooted at \(\bar{u}\) in the hierarchy of \(A\) and \(|A|\) is the number of leaves in the hierarchy of \(A\).

**Definition 4.** The NCP of a record \(v\), a cluster \(C\), and an RT-dataset \(D\), is defined as:

\[
\text{NCP}(r) = \sum_{i \in A \setminus \{r\}} w_i \cdot \text{NCP}_A(r[A_i]),
\]

\[
\text{NCP}(C) = \sum_{r \subseteq C} \text{NCP}(r) \quad \text{and} \quad \text{NCP}(D) = \frac{\sum_{r \subseteq D} \text{NCP}(r)}{|D|}
\]

respectively, where \(w_i \in [0, 1]\) is a weight that measures the importance of a demographic attribute \(A_i\), \(i \in [1, l]\) and is specified by data owners, \(r[A_i]\) denotes the projection of the record \(r\) on the attribute \(A_i\), and \(|D|\) is the size (number of records) of the dataset.

The NCP measure takes values in the \([0, 1]\) range. Lower values of NCP indicate lower data distortion and are preferable. Example 3 below illustrates the computation of the NCP measure.

**Example 3.** Consider the record 6 in the anonymized RT-dataset of Fig. 3. The NCP of the generalized value \(44:47\) in the numerical demographic attribute Age is \(\frac{5}{7} - \frac{2}{7} = 0.094\). In addition, the NCP of the generalized value \(All\) in the categorical demographic attribute Origin is \(\frac{5}{8}\) because the number of leaves of the subtree rooted at \(All\), in the hierarchy of Fig. 4, is 8, and the number of leaves in the same hierarchy is also 8. Similarly, the NCP of the generalized value \(All\) in the categorical demographic attribute Gender is \(\frac{5}{8}\). Thus, assuming a weight \(\frac{1}{8}\) for each demographic attribute, the NCP of the record 6 is \(\frac{1}{8} \cdot 0.097 + \frac{1}{8} \cdot 1 + \frac{1}{8} \cdot 1 = 0.7\). The NCP of the record 7 is also 0.7, and thus the NCP of the cluster \(\{6, 7\}\) is \(\frac{2 \cdot 0.7}{1} = 1.4\). Last, the NCP value of the anonymized dataset in Fig. 3 is \(\frac{4.28}{8}\), where 4.28 is the sum of the NCP values of all records in the dataset.

For diagnosis codes, we use the UL (Utility Loss) [46] measure, which captures the information loss caused by generalization and suppression and it is suitable to use together with generalization models that replace diagnosis codes with sets [46], such as our generalization model (see Definition 2). The following definitions explain the UL measure.

**Definition 5.** Given a generalized diagnosis code \(\bar{u}\), which has replaced \(\subseteq\) diagnosis codes in the RT-dataset, and a weight \(w(\bar{u}) \in [0, 1]\), which reflects the importance of the diagnosis codes replaced by \(\bar{u}\) and is specified by data owners, the UL of \(\bar{u}\) is defined as:

\[
\text{UL}(\bar{u}) = (2^{|\bar{u}|} - 1) \cdot w(\bar{u}).
\]

**Definition 6.** The UL of a record \(r\), a cluster \(C\), and an RT-dataset \(D\), is defined as:

\[
\text{UL}(r) = \frac{\sum_{\bar{u}\subseteq D} \text{UL}(\bar{u}) \cdot \sigma(r)}{2^{|r|} - 1}, \quad \text{UL}(C) = \sum_{r \subseteq C} \text{UL}(r) \quad \text{and} \quad \text{UL}(D) = \sum_{r \subseteq D} \text{UL}(r) / |D|
\]

respectively, where \(\sigma(r)\) computes the total number of diagnosis codes that appear in generalized diagnosis codes in \(r\), \(\sigma(r)\) computes the number of diagnosis codes that have been suppressed from \(r\), and \(|D|\) denotes the size (number of records) of the dataset.

The UL measure takes values in the \([0, 1]\) range. Lower values of UL indicate lower data distortion and are preferable. Example 4 below illustrates the computation of the UL measure.

**Example 4.** Consider the record 6 in the anonymized RT-dataset of Fig. 3 and assume that the weight of each generalized diagnosis code is 1. The UL of the generalized diagnosis code (494.1) is \((2^3 - 1) \cdot 1 = 0\) and that of (458.1, 458.21) is \((2^2 - 1) \cdot 1 = 3\). This reflects that no information loss is incurred when 494.1 is generalized to (494.1), since the generalized diagnosis code is interpreted as the diagnosis code 494.1. Furthermore, it reflects that (458.1, 458.21) incurs more information loss than (494.1), which is expected because (458.1, 458.21) is interpreted as 458.1 and/or 458.21. In addition, 3 diagnosis codes (i.e., 494.1, 458.1, and 458.21) appear in the generalized diagnosis codes in the record 6, and no diagnosis codes have been suppressed. Thus, the UL of the record 6 is \(0.43\). Similarly, the UL of the record 7 is 0.43, and thus the UL of the cluster \(\{6, 7\}\) is \(0.43 = 0.86\). The UL of the dataset in Fig. 3 is \((3 \cdot \frac{2^3 - 1}{15} + 2^2 - 1 + 2^2 - 1 + 0 + 1) / 8 = 0.441\).
To measure the accuracy of answering aggregate queries using anonymized data, we employ the ARE (Average Relative Error) measure [39]. In particular, we consider COUNT queries that model case count studies and have the following SQL-like form:

```
SELECT COUNT(*)
FROM D (or T)
WHERE R and T are supported by a record r in D (or in D)
```

where \( R \) (respectively, \( T \)) is a set of values in demographic attributes (respectively, a set of diagnosis codes), and \( D \) and \( T \) is the RT-dataset before and after anonymization, respectively. That is, a query asks for the number of patients with certain demographics that are diagnosed with a set of one or more diagnosis codes. A similar setting involving only diagnosis codes was considered in [46,47,70].

Lower ARE values indicate more accurate queries answers and are preferable. The following definition explains how ARE can be computed.

**Definition 7.** For a workload \( \mathcal{W} \) of COUNT queries \( q_1, \ldots, q_n \), and for two functions \( C_A \) and \( C_0 \), which count the number of records answering a query \( q_i, i \in [1, n] \) on the anonymized dataset \( D \) and on the original dataset \( D \) respectively, the ARE measure is defined as:

\[
ARE(\mathcal{W}) = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{C_A(q_i) - C_0(q_i)}{C_0(q_i)} \right|
\]

Thus, ARE is computed as the mean error of answering all queries in the query workload \( \mathcal{W} \). Clearly, a zero ARE implies that the anonymized dataset \( D \) is as useful as the original dataset \( D \) in answering the queries in \( \mathcal{W} \) and low values in ARE are preferred. Note that anonymized datasets that have low ARE values are useful not only for answering COUNT queries but also for various analysis tasks that are based on such queries, such as frequent itemset mining and classification [40].

### 3.3. Utility constraints for RT-datasets

Minimizing a data utility measure, such as NCP or ARE, is important. However, it does not guarantee that the anonymized dataset will be useful in intended analysis tasks, as explained in Section 1. This has been recognized by prior works on anonymizing diagnosis codes [46,50], which proposed the concept of utility constraints (see Section 2.2).

In our work, we extend this concept to RT-datasets. Intuitively, a utility constraint models the least preferable way in which a record can be generalized, in the sense that applying more generalization yields the record useless for performing the intended task (i.e., the task is performed with lower accuracy after anonymization).

**Definition 8.** A utility constraint \( u \) is an ordered set \( \{v_{A_i}, \ldots, v_{A_k}, v_{A_m}\} \) where \( v_{A_i} \), \( i \in [1, l + 1] \) is defined as follows:

- If \( A_i \) is a numerical demographic attribute, \( v_{A_i} \) is a range of values of \( A_i \).
- If \( A_i \) is a categorical demographic attribute, \( v_{A_i} \) is a node in the hierarchy of \( A_i \).
- If \( A_i \) is the set-valued attribute, \( v_{A_i} \) is a set of diagnosis codes.

An example of a utility constraint is shown in Table 3. This utility constraint is comprised of the range [40:50] in the numerical demographic attribute Age, the nodes Europe and All in the hierarchy of the categorical demographic attribute Origin and Gender, respectively (see Fig. 4), and the set of the diagnosis codes 053.20 and 053.70, in the set-valued attribute.

<table>
<thead>
<tr>
<th>Table 3</th>
<th>An example of a utility constraint.</th>
</tr>
</thead>
<tbody>
<tr>
<td>( u = {[40:50], \text{Europe}, \text{All}, 053.20, 053.70} )</td>
<td></td>
</tr>
</tbody>
</table>

Our definition of a utility constraint does not pose any restriction in the choice of hierarchy nodes, ranges, or set of diagnosis codes. However, such restrictions are necessary when there are multiple utility constraints, to avoid conflicts between utility constraints which can render the anonymized data useless for the intended task. For example, consider the utility constraint \( u \) in Table 3 and another utility constraint \( u' = \{[19:51], \text{Germany}, \text{Female}, 053.20\} \). Clearly, there is a conflict between these two utility constraints, because \( u \) states that the least acceptable way to generalize a value in Age is to replace it with [40:50], whereas \( u' \) states that replacing the value with [19:51] (i.e., further generalizing the value) is still acceptable. To avoid such conflicts, we require the set of utility constraints to satisfy all of the following requirements:

- For each numerical demographic attribute, any two utility constraints should have the same range or two disjoint ranges.
- For each categorical demographic attribute, any two utility constraints should have the same node or two nodes that have no common leaf-level descendant in the hierarchy of the attribute.
- For the set-valued attribute, any two utility constraints should have the same set of diagnosis codes or two disjoint sets of diagnosis codes.

In the following, we assume that these requirements are satisfied by the specified utility constraints. In addition, we require the specified utility constraints to cover each value in an RT-dataset. More specifically, we require:

- For each numerical demographic attribute \( A_i \), each respective value to be contained in the element (range) \( v_{A_i} \) of a utility constraint.
- For each categorical demographic attribute \( A_i \), each respective value to be contained in the subtree of the hierarchy of \( A_i \) that is rooted at the element \( v_{A_i} \) of a utility constraint.
- For the set-valued attribute \( A_i \), each diagnosis code to be contained in the element \( v_{A_i} \) of a utility constraint.

When the specified utility constraints cover each value in an RT-dataset, the value can be generalized by our algorithm, in a way that guarantees data utility for the intended analysis task, as will be explained later. When there are no specific requirements for certain attribute values, a utility constraint with a “coarse” element (e.g., All) can be specified for these values. For example, if a case count study does not require distinguishing patients according to their Gender, all utility constraints can have the node All in their element for Gender. In the following, we assume that the specified utility constraints cover each value in the given RT-dataset.

The set that contains all the utility constraints is defined as the utility constraint set and is denoted with \( \mathcal{U} \). We now explain when an anonymized RT-dataset satisfies a given utility constraint set.

**Definition 9.** An anonymized dataset \( D \) satisfies a given utility constraint set \( \mathcal{U} \) when, for each record \( r \) of \( D \), there is a utility constraint \( u \) in \( \mathcal{U} \) such that the following conditions hold:

- For each numerical demographic attribute \( A_i \), the generalized value of \( r \) in \( A_i \) is contained in the element \( v_{A_i} \) of \( u \).
- For each categorical demographic attribute \( A_i \), the generalized value of \( r \) of \( A_i \) is contained in the subtree that is rooted at \( v_{A_i} \) in the hierarchy of \( A_i \), where \( v_{A_i} \) is the element of \( u \).
For the set-valued attribute $A_{k,1}$, and for each generalized diagnosis code of $r$, the diagnosis codes that appear in the generalized diagnosis code are contained in the element $v_{k,1}$ of $u$, or of another utility constraint in $U$. Thus, when a utility constraint set is satisfied, generalization does not reduce the utility of the $RT$-dataset for performing the intended analysis task, because no generalized value or generalized diagnosis code exceeds the maximum allowable level of generalization that is dictated by the specified utility constraints.

Example 5 below illustrates when a utility constraint set is satisfied.

Example 5. The anonymized $RT$-dataset of Fig. 3 satisfies the utility constraint set $U = \{u_1, u_2, u_3, u_4\}$, shown in Table 4, because for each record of the anonymized dataset, its generalized values in Age, Origin, and Gender are contained in elements of the same utility constraint and the diagnosis codes that appear in each generalized diagnosis code are contained in a single utility constraint. For example, the generalized values $[19,30]$, Europe, and All of record 1 are all contained in $u_1$ (or equivalently in $u_2$ or $u_3$, since these two utility constraints have the same elements in $\{ \text{Age}, \text{Origin}, \text{Gender} \}$ with $u_1$), and the diagnosis codes 053.20 and 053.71, which appear in the generalized diagnosis code (053.20,053.71) are contained in the same utility constraint $u_2$.

Note that Definition 9 does not capture the impact of suppressing diagnosis codes on data utility. Thus, the utility constraint set in Example 5 is satisfied when 494.1 has been suppressed from record 5. To control the impact of suppression on diagnosis codes, our problem restricts the allowable number of diagnosis codes that can be suppressed, as we will discuss in Section 3.4.

3.4. Problem statement

We now formally define the problem that we aim to solve in this work, as follows.

Problem 1. Given an $RT$-dataset $D$, a utility constraint set $U$, parameters $k$ and $m$, and thresholds $\delta$ and $\epsilon$, construct a $(k,m)$-anonymous version $\tilde{D}$ of $D$, such that all following requirements hold:

1. **Requirement 1.** $\text{NCP}(\tilde{D}) \leq \delta$.
2. **Requirement 2.** $\text{UL}(\tilde{D})$ is minimum.
3. **Requirement 3.** $\tilde{D}$ satisfies the utility constraint set $U$.
4. **Requirement 4.** At most $\epsilon$ diagnosis codes have been suppressed to construct $\tilde{D}$.

Solving Problem 1 is far from trivial for three reasons. First, the problem is computationally difficult (NP-hard), even when the utility constraint set $U$ does not limit the generalization of values (i.e., it contains a utility constraint with the element All for each demographic attribute and the set of all ICD-9 codes for the set-valued attribute) [61]. Second, the requirement to satisfy the utility constraint set, which we introduce in this paper, calls for new methods that are able to take into account the allowable ways of generalizing values. Third, minimizing the information loss in both demographic attributes and diagnosis codes together is difficult. Intuitively, this is because minimizing $\text{NCP}$ calls for creating small groups of records, whereas minimizing $\text{UL}$ calls for creating large groups of records, as explained in [61]. Thus, there is a trade-off between the information loss incurred by generalization on demographic attributes and on diagnosis codes. To control this trade-off, we use the thresholds $\delta$ and $\epsilon$, which are set by data owners.

Currently, there are no algorithms for solving Problem 1.

4. Anonymization methodology

To address Problem 1, we propose the $\text{Art}_{\text{UC}}$ algorithm. In the following, we explain the operation of $\text{Art}_{\text{UC}}$ (Section 4.1) and illustrate its use with an example (Section 4.2). In addition, we propose an adaptation of $\text{Art}_{\text{UC}}$ to help the data owner decide the maximum allowable information loss in the demographic attributes and the maximum allowable suppressed diagnosis codes (Section 4.3). The adaptation is based on progressively relaxing Requirements 1 and 4 of Problem 1.

4.1. The $\text{Art}_{\text{UC}}$ algorithm

The $\text{Art}_{\text{UC}}$ algorithm works in four phases:

1. **Record grouping:** In this phase, groups of records are created. Each group contains all records whose values in the set of demographic attributes are contained in the same utility constraint.
2. **Cluster formation:** In this phase, each group is further split into clusters, which become $k$-anonymous, with respect to the demographic attributes, with minimal $\text{NCP}$.
3. **Cluster merging:** The clusters created in the cluster formation phase are merged. The objective is to create clusters whose diagnosis codes can be anonymized with low information loss, without violating Requirement 1 of Problem 1.
4. **$(k,m)$-anonymization:** Each cluster becomes $(k,m)$-anonymous, by generalizing diagnosis codes. The generalization is performed in a way that aims to satisfy the Requirements 2, 3, and 4 of Problem 1.

After that, the anonymized dataset, which is comprised of the $(k,m)$-anonymous clusters, is returned, and the algorithms terminates. We now describe the pseudocode of $\text{Art}_{\text{UC}}$ in more detail.

**Record grouping.** The record grouping phase is performed in Step 1. In this phase, the records of the $RT$-dataset are placed into different groups, and each record belongs to one group that is formed around a different utility constraint. That is, the values of all records in the group in a numerical (respectively, categorical) demographic attribute $A_i$ are contained in the element $v_{A_i}$ (respectively, in the subtree of the hierarchy of $A_i$ that is rooted at the element $v_{A_i}$) of the utility constraint, for each attribute $A_i \in \{1,\ldots,\ell\}$. The specification of the utility constraint set (see the requirements for the specified utility constraints in Section 3.3) guarantees that each record belongs to at least one group. Typically, the created groups contain more than $k$ records.

This is reasonable since $k$ is a small constant in practice. In the extreme case where a group has fewer than $k$ records, generalization alone cannot enforce $(k,m)$-anonymity. Thus, to achieve this goal, the algorithm suppresses the records of the group (Step 2).

In our experimental analysis (Section 5), we demonstrate that such suppressions are extremely rare for a wide range of parameter settings and utility constraint sets. Specifically, $\text{Art}_{\text{UC}}$ executed Step 2 only for $k = 100$ (20 times larger than the typical value) and for only one utility constraint. As a result, fewer than 0.01% of the input records were suppressed.

### Table 4

A utility constraint set.

<table>
<thead>
<tr>
<th>$u_1$</th>
<th>$u_2$</th>
<th>$u_3$</th>
<th>$u_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$[19, 50]$, All, All, [458.1, 458.21]</td>
<td>$[19, 50]$, All, All, [053.20, 053.70]</td>
<td>$[19, 50]$, All, All, [494.1]</td>
<td>$[51]$, Africa, Male, [493.2]</td>
</tr>
</tbody>
</table>
Algorithm 1. ARTUC

Algorithm: ARTUC
Input: An $RT$-dataset $D$, a utility constraint set $U$, and anonymization parameters $k$, $m$, $\delta$, and $\epsilon$.
Output: A $(k, k^m)$-anonymous $RT$-dataset $D$ that corresponds to $D$, satisfies the Requirements 1, 2, and 4 of Problem 1, and has minimal $UL$.

// Record grouping
1. Group the records of $D$ into a set $G$ of $|U|$ groups such that the records of each group are contained in the same utility constraint of $U$.
2. Remove all groups from $G$ that have less than $k$ records.

// Cluster formation
3. Let $C$ be an empty set of clusters.
4. foreach group $G \in G$ do // Form $k$-sized clusters
   5. while group $G$ has more than $k$ records do
      6. Move a random record from group $G$ into a new cluster $C$.
      7. while $C$ has fewer than $k$ records do
         8. Find the record $r_{NCP} \in G$ that minimizes the NCP of $C \cup \{r_{NCP}\}$ when generalized with Definition 2.
         9. Move record $r_{NCP}$ from group $G$ into cluster $C$.
      10. Add cluster $C$ into the set of clusters $C$.
   11. foreach group $G \in G$ do // Consider the groups with records that are not assigned in clusters
      12. foreach record $r$ in group $G$ do
         13. Find the cluster $C_{NCP}$ in $C$ that minimizes the NCP of $C_{NCP} \cup \{r\}$ when generalized with Definition 2.
         14. Move $r$ from group $G$ into cluster $C_{NCP}$.
   15. Apply the generalization of Definition 2 to each cluster in the set of clusters $C$.
      if $NCP(C) > \delta$ then return false ;

// Cluster merging
17. Merge all clusters of $C$ with the same generalized values in all the demographic attributes.
18. repeat
19. Select the cluster $C_{UL} \in C$ with minimum $UL$ when generalized with our adaptation of the CBA algorithm.
20. Search for a cluster $C'_{UL} \in C - C_{UL}$ such that (a) $C_{UL} \cup C'_{UL}$ has minimum $UL$ (when generalized with our adaptation of the CBA algorithm), and (b) $NCP(C - C_{UL} - C'_{UL})$ is at most $\delta$ (when generalized with Definition 2).
21. if $C_{UL}$ exists then Remove from $C$ clusters $C_{UL}$ and $C'_{UL}$ and add cluster $C_{UL} \cup C'_{UL}$;
22. until the set of clusters $C$ does not change;

// $(k, k^m)$-anonymization
23. Let $C$ be an empty set of clusters that will store the anonymized result.
24. foreach cluster $C$ in $C$ do
25. Anonymize $C$ using our adaptation of the CBA algorithm and add to $C$ result.
26. if the total number of suppressed diagnosis codes exceeds $\epsilon$ then return false ;
27. return $D$

Cluster formation. The cluster formation phase is performed in Steps 3 to 16. In each iteration, a different group $G \in G$ is considered and is divided into clusters, each containing $k$ records. This is performed by a heuristic that adds into the cluster the record that incurs the minimum increase to the NCP value of the cluster, after generalization, until the cluster contains $k$ records (Steps 6–10). Note that NCP is measured for the cluster, after the generalization of all its values in the demographic attributes. The generalization is performed by applying Definition 2 to each demographic attribute. Note also that the NCP value of the cluster cannot decrease, after the record addition (see Definition 3). Each created cluster is also saved into the set of clusters $C$ (Step 10).

Next, the algorithm considers the records that have not been added into clusters (Steps 11–14). Clearly, in every group, there can be at most $k - 1$ such records; if there were more, they would have formed a cluster. Each of these records is moved into the cluster that fits best, in the sense that its addition incurs the lowest increase in the NCP of the cluster. Then, the demographics of each cluster are generalized using Definition 2 (Step 15). Following, the algorithm checks if Requirement 1 of Problem 1 is satisfied (Step 16). Specifically, if the NCP of the resultant clustering $C$ exceeds the parameter $\delta$, then the algorithm terminates, since an acceptable solution in terms of the information loss with respect to the demographic attributes cannot be found.

Cluster merging. The cluster merging phase is performed in Steps 17–22. First, ARTUC combines clusters which have the same generalized values in all demographic attributes (Step 17). Clearly, the resultant clusters are still $k$-anonymous, and the NCP of the set of clusters $C$ does not change (see Definition 4). However, the merged clusters offer more room for minimizing the information loss of the set-valued attribute. This is because they contain more diagnosis codes and are more likely to become $k^m$-anonymous with respect to the diagnosis codes, with lower information loss according $UL$.

Subsequently, ARTUC merges together clusters with semantically close diagnosis codes aiming at minimizing $UL$ (Steps 18–22). To this end, the algorithm first selects the cluster $C_{UL}$ that has the minimum $UL$ when the diagnosis codes are anonymized (Step 19). The anonymization is performed by an adaptation of the CBA algorithm [45] (see Section 2.2). Our adaptation enforces $k^m$-anonymity to a cluster of records, instead of preventing re-identification based on the specified sets of diagnosis codes as CBA does (a detailed explanation is provided in Appendix A). Following, in Step 20, ARTUC searches for a cluster $C'_{UL}$ which, if merged with $C_{UL}$, results in a cluster that has:
1. minimum UL (when generalized with our adaptation of the CBA algorithm), and
2. NCP at most \( \delta \) (when generalized with Definition 2).

Intuitively, \( C_{\text{uc}} \) is the "closest" cluster to \( C_{\text{uc}} \) with respect to UL that does not violate the NCP bound set by Requirement 1 of Problem 1. If \( C_{\text{uc}} \) exists, the two clusters are merged and the algorithm attempts to merge another pair of clusters. Otherwise, the cluster merging stops.

\((k, k^n)\)-anonymization. The \((k, k^n)\)-anonymization phase is performed in Steps 24–27. In this phase, \( \text{ArtUC} \) generalizes the diagnosis codes in a cluster, using our adaptation of the CBA algorithm (Step 25), and checks whether the total number of suppressed diagnosis codes exceeds the parameter \( \epsilon \). In this case, the utility constraint set cannot be satisfied, and the algorithm terminates (Step 26). Otherwise, the cluster is added into an initially empty set \( D \) that stores the output, and the next cluster is considered.

After all clusters have been considered, \( \text{ArtUC} \) returns the set \( D \), which is comprised of all clusters and is a \((k, k^n)\)-anonymized version of the RT-dataset \( D \) (Step 27).

\( \text{ArtUC} \) is efficient and scales well with respect to the dataset size and anonymization parameters. A time complexity analysis of our algorithm can be found in Appendix C.

4.2. Example using \( \text{ArtUC} \)

To illustrate the operation of the \( \text{ArtUC} \) algorithm, we apply it to enforce \((2, 2^2)\)-anonymity on the dataset of Fig. 1a using the utility constraint set of Table 4. We set the parameters of \( \text{ArtUC} \) as follows: \( k = 2, m = 2, \delta = 0.6, \) and \( \epsilon = 2.2 \).

During the record grouping phase, the records of the dataset are split into the groups \( G_1 = \{0, 1, 2, 3, 6, 7\} \) and \( G_2 = \{4, 5\} \) (Step 1). \( G_1 \) is formed around the utility constraint \( u_1 \) (or equivalently around \( u_2 \) or \( u_3 \), which have the same elements in \{Age, Origin, Gender\} with \( u_1 \)), while \( G_2 \) is formed around \( u_4 \). Next, \( \text{ArtUC} \) considers \( G_1 \) and creates the first cluster, \( C_1 \) (Steps 4–10). Let us assume that the record \( 0 \) is selected and moved to \( C_1 \), in Step 6. Then, the record 1 is added into \( C_1 \), because generalizing the demographic values of the records 0 and 1 together results in the minimum NCP value (Steps 7–9). Similarly, \( \text{ArtUC} \) creates the clusters \( C_2 \) and \( C_3 \), which contain the records \( \{2, 3\} \) and \{6, 7\} of \( G_1 \), respectively.

After that, the algorithm considers the second group, \( G_2 \), and creates the cluster \( C_4 \) (Steps 4–5). Since all records of the groups are assigned to clusters, Steps 11–14 are skipped. Then, \( \text{ArtUC} \) generalizes the demographic attributes of each cluster using Definition 2 (Step 15), as shown in Fig. 1a, and proceeds into the next phase, because the NCP of the set of clusters \( C = \{C_1, C_2, C_3, C_4\} \) is lower than \( \delta \) (Step 16).

In the cluster merging phase, the cluster \( C_1 \) has the minimum UL when generalized (Step 19). Then, the algorithm finds the cluster \( C_2 \), which satisfies both conditions (i.e., \( (a) C_1 \cup C_2 \) has minimum UL, and \( (b) \) NCP of \( C = C_1 \cup C_2 \) is at most \( \delta \) (Step 20) and merges \( C_1 \) and \( C_2 \) (Step 21). Subsequently, \( \text{ArtUC} \) proceeds into the following phase, because any further merging of clusters results in a set of clusters having a higher NCP than \( \delta \) (i.e., violates the Requirement 1 of Problem 1).

In the \((k, k^n)\)-anonymization phase (Steps 24–26), the diagnosis codes in each cluster are generalized, and the \((k, k^n)\)-anonymized RT-dataset, shown in Fig. 3, is returned (Step 27).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>D</th>
<th># of demographics</th>
<th># of distinct diagnosis codes</th>
<th>Max. Avg. # diagnosis codes per record</th>
</tr>
</thead>
<tbody>
<tr>
<td>EHRD</td>
<td>208,387</td>
<td>2</td>
<td>13,963</td>
<td>185, 16.21</td>
</tr>
<tr>
<td>INFORMS</td>
<td>36,553</td>
<td>5</td>
<td>619</td>
<td>17, 4.27</td>
</tr>
</tbody>
</table>

### Table 5
Description of EHRD and INFORMS dataset.

### Table 6
Utility constraint sets used in our experiments with (a) the EHRD dataset, and (b) the INFORMS dataset. In each utility constraint set, the interval [\( [\) is the "closest" cluster to \( \{001, 002, 003, \ldots \} \) in the ICD-9 taxonomy.

### 4.3. Adaptation of \( \text{ArtUC} \) based on progressive relaxation of Requirements 1 and 4

As can be seen in the pseudocode, our algorithm produces an anonymized dataset that satisfies Requirements 1, 3, and 4 of Problem 1 and has minimal UL, except in the following cases:

- the NCP of the anonymized dataset with respect to the demographics exceeds the specified \( \delta \) (i.e., Requirement 1 is not satisfied), or
- the number of suppressed diagnosis codes exceeds the specified value of \( \epsilon \) (Requirement 4 is not satisfied).

In these cases, the satisfaction of the requirements is treated as a hard constraint, and the algorithm does not produce an anonymized dataset. This algorithmic design choice clearly reflects the semantics of utility constraints, which dictate that a group of records that is "too" generalized is useless for analysis, and it is consistent with prior work [46–48].

An alternative design choice is to relax Requirements 1 and 4 of Problem 1 progressively, as the algorithm is being executed, and let the data owner decide when the produced anonymized dataset is useful for them (i.e., it has acceptable NCP a sufficiently low number of suppressed diagnosis codes). This adaptation of \( \text{ArtUC} \) avoids the need for specifying \( \delta \) and \( \epsilon \) a priori and turns \( \text{ArtUC} \) into an "any time" method, in the spirit of the method of [20]. The adaptation is straightforward, and it is performed by replacing the Steps 16–25 of \( \text{ArtUC} \) with the pseudocode provided in Algorithm 2 below.
Algorithm 2. Adaptation of ARTUC that progressively relaxes Requirements 1 and 4 of Problem 1.

```
1 Anonymize each cluster in C using our adaptation of the CBA algorithm
2 \(D \leftarrow \) the set of anonymized clusters in C
3 if \(NCP(C)\) is not deemed acceptable by the data owner then
   return false
4 else
5   Merge all clusters of C with the same generalized values in all the demographic attributes
6   repeat
7     Select the cluster \(C_{UL} \in C\) with minimum \(UL\) when generalized with our adaptation of the CBA algorithm.
8     Search for a cluster \(C_{UL}' \in C - C_{UL}\) such that \(C_{UL} \cup C_{UL}'\) has minimum \(UL\) (when generalized with our adaptation of the CBA algorithm)
9     if \(C_{UL}\) exists then
10        temp \(\leftarrow C - C_{UL} \cup \{C_{UL} \cup C_{UL}'\}\)
11        Apply the generalization of Definition 2 to each cluster in temp
12        Anonymize each cluster in temp using our adaptation of the CBA algorithm
13        if \(NCP(\text{temp})\) is deemed acceptable by the data owner then
14           Remove from C clusters \(C_{UL}\) and \(C_{UL}'\) and add cluster \(C_{UL} \cup C_{UL}'\)
15           Update \(D\)
16        else if the number of suppressed diagnosis codes is acceptable then
17           return \(D\)
18        else
19           return false
20   until the set of clusters \(C\) does not change;
21 return \(D\)
```

As can be seen in the pseudocode of Algorithm 2, an anonymized dataset \(D\), comprised of all the formed clusters, is created (steps 1–2), and the data owner is asked whether the \(NCP\) of \(D\) is sufficiently low (step 3). If not, \(false\) is returned (step 4), since subsequent cluster merging cannot decrease \(NCP\), as mentioned above. Otherwise, the adaptation proceeds into the cluster merging phase (steps 5–21). In this phase, the clusters with the same generalized values in all demographic attributes are combined (step 6), as in ARTUC. Then, the clusters are merged iteratively (steps 7–21). Differently from ARTUC, however, each cluster, including the temporarily merged cluster \(C_{UL}\), is \((k, k^m)\)-anonymized (steps 12–13), and the data owner is asked whether the \(NCP\) of the resultant dataset is sufficiently low (step 14). If it is, the clusters and dataset are updated (steps 15–16), and we proceed into the next iteration. Otherwise, the \(NCP\) is deemed unacceptable, and the data owner is asked whether the number of suppressed diagnosis codes is sufficiently low (step 17). If it is, the clusters are not merged and an anonymized dataset with acceptable \(NCP\) and number of suppressed diagnosis codes is returned (step 18). Otherwise, \(false\) is returned, since further cluster merging and \((k, k^m)\)-anonymization can only increase the number of suppressed diagnosis codes (steps 19–20). The cluster merging continues as long as the \(NCP\) is acceptable and the set of created clusters does not change (step 21). After that, the anonymized dataset, which has acceptable \(NCP\) and number of suppressed diagnosis codes, is returned (step 22).

5. Experimental evaluation

In this section, we evaluate ARTUC in terms of data utility preservation and efficiency. We compare ARTUC with two anonymization algorithms that are applicable to RT-datasets, namely \(RSG\) and BASELINE. \(RSG\) [61] aims to minimize the overall information loss without considering the intended analysis requirements (see Section 2.1). BASELINE is a baseline method, which performs record grouping and \((k, k^m)\)-anonymization. BASELINE is similar in principle to the algorithm of Takahashi et al. [68], in that it pre-generalizes the demographic attributes. The BASELINE algorithm works as follows. First, it creates a set of groups of records, each of which corresponds to a different utility constraint. Then, it considers each group and generalizes the demographic attributes and diagnosis codes of every record in the group. The demographic attributes are generalized as specified by the utility constraint corresponding to the group (i.e., the value in each attribute is replaced by the corresponding value of the utility constraint of the group), while the diagnosis codes are anonymized following the Steps 24–26 of Algorithm ARTUC. BASELINE outperforms the algorithm of [68] in terms of preserving data utility, because it employs \((k, k^m)\)-anonymity instead of \(k\)-anonymity and set-based generalization instead of hierarchy-based generalization for diagnosis codes.

5.1. Experimental setup

We have implemented all algorithms in C++ and ran all experiments on an Intel i7 at 3.2 GHz with 32 GB of RAM running Mac OS X 10.8. In our experiments, we use two RT-datasets, namely EHRD and Informs. Each record in these datasets contains demographics and ICD-9 codes. The EHRD dataset is proprietary (provided by a university medical center). The Informs dataset is publicly available at https://sites.google.com/site/informsdataminigcontest/data. The processing and analysis of these datasets was performed in a privacy-preserving way, according to standard practices and policies. Table 5 summarizes the characteristics of the EHRD and Informs datasets.

In our experiments, the default anonymization values were set as follows:

\[
k = 10, \quad m = 2, \quad \delta = 0.04, \quad \text{and} \quad \epsilon = 0.06 \cdot |\text{distinct codes}|
\]

The hierarchies were constructed as in [72]. To experiment with different types of utility requirements, we use 9 different utility constraint sets (illustrated in Table 6). The default utility constraint set for EHRD is \(UC_1\) (see Table 6a). For instance, each utility constraint in this utility constraint set is comprised of:
The ICD-9 taxonomy organizes diagnosis codes into more general, semantically related concepts. The ICD-9 taxonomy for the diagnosis codes that are contained in the dataset of Fig. 1a is depicted in Fig. 6. The leaves contain the most detailed diagnosis codes, which have 4 or 5 numerical digits, and the immediate ancestors of leaves are 3-digit diagnosis codes. For example, the diagnosis code 053.20 (Herpes zoster dermatitis of eyelid) is a leaf, whose immediate ancestor is the 3-digit diagnosis code 053 (Herpes zoster). In turn, the immediate ancestors of 3-digit codes are Sections and those of Sections are Chapters. For example, the immediate ancestor of 053 is the Section 050–059 (Viral diseases accompanied by exanthem), and the immediate ancestor of the latter is the Chapter 001–139 (Infectious and parasitic diseases). Last, the immediate ancestor of all Chapters is the root value of the taxonomy, which represents any ICD-9 diagnosis code.

Note that the utility constraint sets UC1 and UC3 in Table 6a contain the same elements in the demographic attributes but different elements in the set-valued attribute. Furthermore, Chapter1 of the utility constraint u1 in UC1 contains all the diagnosis codes in {001–139} of u1 in UC2, which represents the codes with the immediate ancestor 001. Thus, UC1 is less restrictive than UC2 and is expected to be satisfied with less generalization. For the same reason UC1 is less restrictive than UC2, which is less restrictive than UC3. Similar relations exist among UC4 to UC6 and among UC7 to UC9 in Table 6a. In addition, it is easy to see that u1 in UC1 is less restrictive than u1 in UC2, because its element {0: 50} in Age contains the element {0: 12} of the utility constraint u1 in UC1.

To evaluate data utility, we use the NCP, UL, and ARE measures (see Section 3.2). Lower values of these measures indicate lower data distortion and are preferable. Note that none of the methods optimizes ARE directly, since the methods do not take as input the query workloads. We use workloads of 100 queries, involving demographics and/or diagnosis codes, which retrieve random values and/or sets of 2 diagnosis codes by default, following [39,47]. Since ARE reflects the average number of records that are retrieved incorrectly as part of answering a workload of queries, low ARE scores imply that anonymized data can be used to estimate the number of patients having certain demographic values and diagnosis codes fairly accurately.

5.2. Data utility comparison

We first evaluate all methods, with respect to the NCP and the UL measure, for varying δ. The results are reported in Figs. 7 and 8. Increasing δ leads both ARTUC and RMK to create larger clusters, which have higher NCP and lower UL. On the other hand, BASELINE is not affected by δ, because it creates clusters of fixed size, which favor the anonymization of diagnosis codes with low UL. In more detail, ARTUC outperformed BASELINE in NCP when we used EHRD dataset, achieving lower scores by 620% on average, while its UL scores were higher by 240% on average. The results for the INFORMS dataset, shown in Fig. 8, are qualitatively similar to those of Fig. 7. Specifically, ARTUC achieved lower scores by 662.07% on average, while its UL scores were higher by 138.01% on average. This indicates that the pre-generalization of demographic attributes, used by BASELINE, incurs high information loss, unlike the cluster formation and cluster merging strategies that are employed by ARTUC. In addition, the NCP scores of ARTUC and RMK were equal and the UL scores of ARTUC were 50% better on average (and up to 92% better, for the case of EHRD, and 32.24% better on average for the case of INFORMS). This shows that ARTUC is able to satisfy the utility constraint set, while preserving the information in both demographics and diagnosis codes. Moreover, Fig. 7c presents the additional data utility loss incurred by using RMK or BASELINE instead of ARTUC. The additional data utility loss for RMK is computed as:

\[
\frac{1}{2} \left( \frac{\text{NCP}(\text{BASELINE}) - \text{NCP}(\text{ARTUC})}{\text{NCP}(\text{ARTUC})} + \frac{\text{UL}(\text{BASELINE}) - \text{UL}(\text{ARTUC})}{\text{UL}(\text{ARTUC})} \right) \cdot 100\%
\]

The computation assumes that NCP and UL are equally important and is similar for BASELINE. As can be seen in Fig. 7c, the additional data utility loss for RMK is 18.6% on average and that for BASELINE is 252.1% on average. The result in Fig. 8c is similar.

In the following experiments, we evaluate all methods, with respect to the ARE measure. The results are shown in Figs. 9 and 10. For each dataset, we use three query workloads, namely W1, W2, and W3. In summary:

- The queries in W1 retrieve information based on the values of two demographics attributes.
- The queries in W2 retrieve information based on the value of one demographic attribute and one diagnosis code.
- The queries in W3 retrieve information based on two diagnosis codes.

For example, a query in W2, which was used in the experiments with the EHRD dataset, can retrieve the number of patients associated with Age = 32 and Chronic obstructive asthma.

As expected, the effectiveness of the evaluated methods is affected by the query workloads. For example, BASELINE, which favors the anonymization of diagnosis codes, has the worst effectiveness for W1 and the best for W3 (see Fig. 9a and c). ARTUC is more effective than BASELINE for both W1 (e.g., 812% better, on average in the case of EHRD and 736.60% better on average, in the case of INFORMS) and W2 (34% better, on average, in the case of EHRD, and 17.53% better, on average, in the case of INFORMS). These results highlight that our method allows accurately answering queries that involve demographics and diagnosis codes, which are the most important when publishing RT-datasets. On the other hand, RMK merges together clusters that have semantically distant values in the demographic attributes, and this results in high ARE scores, particularly for W2 and W3 (see Fig. 9b and c for EHRD and Fig. 10b and c for INFORMS). For example, note in Fig. 9b and c, that ARTUC is more effective than RMK for W1 (47% better, on average), W2 (123% better, on average), and W3 (14% better on average).

In summary, the results in Figs. 7–10 demonstrate that ARTUC preserves data utility better than BASELINE and RMK. In particular, BASELINE did not permit accurate analysis on demographics, while RMK did not support the intended analysis tasks. Having established that ARTUC is more accurate than BASELINE and RMK, we will only report results for ARTUC in the following sections. In addition, the results using the EHRD dataset are quantitatively similar to those using the INFORMS dataset. Thus, for brevity, we will only report results for EHRD in the following sections.
parameters $k$ on diagnosis codes, using the method to create larger clusters, which favors the anonymization constraint set UC. Specifically, we vary the utility scores, as shown in Fig. 11d. UL, when there is a fixed utility constraint set and varying values of $k, m$ and $\delta$ parameters.

In the first set of experiments, we evaluate the information loss on diagnosis codes, using the UL measure. Specifically, we varied the parameters $k, m, \delta$, as well as the size of the input RT-dataset $|D|$, and present the results with respect to UL in Fig. 11. As can be seen in Fig. 11a and b, UL values increase with $k$ and $m$. Larger values of $k$ and $m$ enforce stricter privacy requirements. To satisfy these requirements, ART applied more generalization which led to higher formation loss.

On the contrary, as illustrated in Fig. 11c, the UL scores are lower, for larger $\delta$ values. This is because a larger $\delta$ leads our method to create larger clusters, which favor the anonymization of diagnosis codes with low information loss. Similarly, increasing the dataset size, resulted in lower UL scores, as shown in Fig. 11d.

5.3 Data utility evaluation of ARTUC (fixed utility constraint set, varying parameters)

In this section, we examine the data utility offered by our method, when there is a fixed utility constraint set and varying anonymization parameters. Specifically, we use the utility constraint set UC of Table 6 and report results for UL and ARE, when we vary the size of the input dataset $|D|$ and the values of parameters $k, m$ and $\delta$ parameters.

In the first set of experiments, we evaluate the information loss on diagnosis codes, using the UL measure. Specifically, we varied
These datasets can be performed with lower information loss. However, these datasets contain all records of the smaller datasets. In the following experiments, we evaluate data utility, using the ARE measure, and the workloads $W_1$, $W_2$, and $W_3$, used in Section 5.2. Specifically, Fig. 12a–c shows the ARE scores, for varying $k$, and for the workloads $W_1$, $W_2$, and $W_3$, respectively. We also report the ARE measure, and the workloads $W$ and $W_1$, which correspond to the minimum and maximum number of clusters constructed by our method, respectively. $NCP$ did not exceed $\delta$ in all tested cases. As can be seen, increasing $\delta$ leads to higher ARE scores for $W_1$ and $W_2$ but to lower ARE scores for $W_3$. This is because, ARTUC creates larger clusters of similar diagnosis codes when $\delta$ is larger, which lead to higher information loss on the demographic attributes but to lower information loss on the diagnosis codes. We also report the ARE scores for varying $m$ in Fig. 14a and b, which correspond to $W_2$ and $W_3$, respectively. Since varying $m$ only affects the information loss on diagnosis codes, we do not report the result for $W_1$. Note that increasing $m$ results in larger ARE scores. This is because more combinations of diagnosis codes need protection when $m$ is larger, and this leads ARTUC to apply more generalization.

5.4. Data utility evaluation of ARTUC (varying utility constraint sets, fixed parameters)

In this section, we examine the data utility offered by our method, when all parameters are fixed to their default values and various utility constraint sets are used. In particular, we study how the specified utility constraint set affects: (i) the percentage of suppressed diagnosis codes, (ii) the $UL$ measure, and (iii) the ARE measure. We do not report results for NCP, because its value was equal to the specified threshold $\delta = 0.04$. We used two different setups of utility constraint sets, which are presented in Table 7. In Setup 1, ARTUC is applied with UC$_1$, UC$_2$, or UC$_3$. In Setup 2, one of the utility constraint sets UC$_1$, UC$_4$, and UC$_5$ is used instead. As we move from left to right in the first row of Table 7, the utility constraint sets become more restrictive. That is, UC$_1$ is the least restrictive set among those in Setup 1 (see Section 5.1), UC$_3$ is the most restrictive, and UC$_2$ lies in between UC$_1$ and UC$_3$. On the contrary, as we move from left to right in the second row Table 7, the utility constraint sets become less restrictive. This is because they contain larger intervals in the demographic attribute Age. In our first experiment, we report the percentage of suppressed diagnosis codes (Fig. 15a). The percentage of suppressed diagnosis codes is the lowest for UC$_1$, the least restrictive utility constraint set in Setup 1, and it is the highest for UC$_3$. This is because restrictive utility constraint sets limit the number of allowable generalizations and force ARTUC to apply suppression, in order to enforce $(k, k^m)$ anonymity. Suppression is employed only when the use of generalization is not sufficient to $(k, k^m)$-anonymize a cluster. Thus, the percentage of suppressed diagnosis codes is relatively small (e.g., 1% for UC$_2$ and 5.09% for UC$_3$) and does not exceed the specified threshold $\epsilon$.

In the following experiment, we consider the $UL$ metric and in Fig. 15b illustrate its values for the utility constraints sets of Setup 1. Note that $UL$ values decrease between UC$_1$ and UC$_2$ and slightly increase between UC$_2$ and UC$_3$. To explain this behavior, we
decompose $UL$ into the utility loss caused by generalization, denoted by $UL_{gen}$, and the utility loss caused by suppression, denoted by $UL_{sup}$ (obviously $UL = UL_{sup} + UL_{gen}$). As can be seen in Fig. 15b, $UL_{gen}$ decreases as the utility constraints become more restrictive (i.e., moving from $UC_1$ to $UC_2$ and $UC_3$), whereas $UL_{sup}$ follows an opposite trend. This justifies that the increase in $UL$ for $UC_3$ is a result of the use of suppression by ArtUC (see also the percentage of suppression that corresponds to $UC_3$ in Fig. 15a).

In Fig. 15c, we use the utility constraint sets of Setup 2. These utility constraint sets were satisfied without suppression, thus, the $UL$ score decreases as the utility constraint set becomes less restrictive (from the most strict $UC_1$ to the more loose $UC_7$).

Subsequently, we report the ARE scores for both setups. Fig. 16a and b correspond to the utility constraint sets of Setup 1 and to the query workloads $W_2$ and $W_1$, respectively. ARE scores are the largest for $UC_1$, the most restrictive utility constraint set $UC_3$ of Setup 1 since to satisfy $UC_1$ more generalization and suppression is required. Similar results are in order for Setup 2; Fig. 16c–e corresponds to the query workloads $W_1$, $W_2$, and $W_3$, respectively. Specifically, $ARE$ is larger for $UC_1$, which is the most restrictive utility constraint set of Setup 2.

In summary, the use of different utility constraint sets affects data utility. For instance, replacing the utility constraint $UC_1$ with $UC_3$, results in higher $UL_{sup}$ and lower $UL_{gen}$ scores. On the contrary, replacing $UC_1$ with $UC_7$, results in high NCP and low $UL$ scores, as less restrictive constraints lead ArtUC to create larger groups of records. However, in both cases, our method preserved utility well, according to all tested measures. Furthermore, our method was able to generate data that can be used to support the intended studies, as the utility constraints were always satisfied.

5.5. Efficiency evaluation of ArtUC

In this section, we evaluate the execution time of our method ArtUC, by varying the parameters $k$, $m$, and $D$ (size of dataset). The results are depicted in Fig. 17a–c. Fig. 17a shows that the runtime of our method decreases with $k$, as larger values of $k$ result in larger initial clusters and fewer clusters merging operations. On
the other hand, Fig. 17b shows that runtime increases with \( m \), as there are more combinations of diagnosis codes that must be protected. Notice though that the increase is subquadratic. Finally, Fig. 17c shows that the execution time of our method increases with \(|D|\). The increase is also subquadratic in \(|D|\). Overall, the results show that our method can anonymize datasets containing thousands of patient records within minutes.

6. Discussion

This section explains how our work can be extended to deal with different types of healthcare data and privacy requirements. In addition, it discusses limitations, which suggest opportunities for further research.

Our approach was developed for RT-datasets in which the set-valued attribute contains diagnosis codes. However, in certain applications, a patient record may also contain procedural codes (e.g., CPT codes which describe medical, surgical, and diagnostic services [55]), as well as medications. Extending our approach to deal with such datasets requires forming a set-valued attribute that contains all different types of codes. More formally, multiple set-valued attributes, \( A_{i1}, \ldots, A_{im} \), can be modeled as a single set-valued attribute \( A_i \), whose domain (set of possible values) contains every value in the domain of \( A_{i1} \) or \( A_{i2} \) or \( \ldots \) \( A_{im} \), preceded by the domain name. For example, the domain of \( A_i \) contains a value \( A_{i1}, u \) to represent the value \( u \) of \( A_{i1} \). This is simply to distinguish between values of different attributes. In addition, each element \( v_{Ai} \) of a utility constraint must contain values of a single attribute, to prevent the generalization of values of different attributes (e.g., diagnosis and procedural codes), which are difficult to interpret.

Furthermore, our approach aims to prevent identity disclosure, which is the most important privacy requirement for healthcare data. This is suggested by the fact that "all the publicly known examples of re-identification of personal information have involved identity disclosure" [15] and by the fact that the majority of healthcare data anonymization methods focus on preventing identity disclosure. However, our algorithm can be extended to prevent attribute disclosure [72], when there are sensitive diagnosis codes with which patients are not willing to be associated. To achieve this, we may appropriately modify our adaptation of the CBA algorithm used in Steps 19, 20 and 25 of Algorithm ART\(_{UC} \) (see also Appendix A) so that it enforces \((k, l^m)\)-diversity [61] on the sensitive diagnosis codes of each cluster. The \((k, l^m)\)-diversity principle guarantees that an attacker who knows all the values of the demographic attributes and up to \( m \) diagnosis codes of a patient cannot associate these diagnosis codes with any combination of sensitive diagnosis codes, with probability larger than \( \frac{1}{l} \), where \( l > 2 \) is a parameter specified by data owners. This modification requires to apply additional generalization to ensure that any combination of \( p \) and sensitive diagnosis codes appears in at least \( l \) records of the anonymized cluster. Thus, this modification incurs additional information loss and computational overhead. The evaluation of the extended version of our algorithm that enforces \((k, l^m)\)-anonymity is left as future work.

Moreover, unlike other data anonymization methods, our approach assumes that data owners are able to select appropriate values for the parameters \( k \) and \( m \), which model the privacy requirements, as well as for \( \delta \) and \( \epsilon \), which model the maximum allowable level of information loss caused by generalizing demographic attributes and by suppressing diagnosis codes, respectively. However, configuring these parameters in an optimal manner, for a given RT-dataset, is not straightforward.

For example, the ART\(_{UC} \) algorithm stops (i.e., the utility requirements are not satisfied) when: (I) the NCP of the anonymized dataset with respect to the demographics exceeds the specified \( \delta \) (Step 16), or (II) the number of suppressed diagnosis codes exceeds the specified \( \epsilon \) (Step 26). In these cases, the satisfaction of the utility requirements (requirements 1 and 3 of Problem 1) is treated as a hard constraint, and thus the algorithm does not produce an anonymized dataset. Our choice clearly reflects the semantics of utility constraints, which dictate that a group of records that is "too" generalized is useless for analysis, and is consistent with prior work [45–47].

An alternative choice would be to relax the utility requirements, by setting larger \( \delta \) and \( \epsilon \). This raises the question of how to relax the requirements in a way that is easy and intuitive for the data owners, given that their specification is data-dependent. To address this question, we propose to involve the data owner in the execution of our algorithm, so that they can see which the information loss with respect to the relational attributes and the number of suppressions are acceptable, as the algorithm progresses. This makes our algorithm an "any time" method, in the spirit of [20]. This can be easily done by the following two adaptations of our algorithm. The first adaptation is to perform generalization after the current step 3. The NCP of this dataset gives the maximum possible \( \delta \) that the user can set. Then, to perform generalization of each cluster after its creation and output the current NCP of the resultant dataset, which will get smaller as more clusters were created. The data owner can stop the cluster formation phase, when the NCP is sufficiently small. After that, the algorithm will continue into cluster merging. The second adaptation is to output the total number of suppressed diagnosis codes after step 12 of GEN\(_{UC} \). The data owner can stop the execution of the algorithm when the number of suppressions is deemed "too" high. As shown in our experiments, the number of suppressions is very small or zero in practice, so the data owner will not have to examine the output many times, which makes the adaptation easy to use. We acknowledge that the data owner may need to examine other data quality indicators in addition to NCP and number of suppressed diagnosis codes. Towards this goal, we aim to incorporate our algorithm, together with the adaptations, in the SECRETA:
anonymization tool [60]. The tool offers a GUI that outputs many data utility indicators that may assist the data owner towards the specification of \( \delta \) and \( \epsilon \).

Our approach aims to produce an anonymized RT-dataset that remains useful for intended analytic tasks, modeled with utility constraints, as well as for general analytic tasks. However, in a different setting, the anonymized dataset needs to remain useful for building a pre-determined data mining model (e.g., a classifier). In this setting, a different anonymization methodology which aims to preserve data utility for the specified data mining model (in the spirit of [53]) may preserve data utility better.

Last, our approach considers an unordered set of diagnosis codes, as the existing algorithms for anonymizing RT-datasets [31,61] do. However, certain applications, such as longitudinal studies, require ordered sets (or ordered multisets) of diagnosis codes [69]. Anonymizing such data in a utility-preserving way is challenging because it requires preserving the sequentiality of data. To the best of our knowledge, the problem has not been considered and serves as an interesting avenue for future work.

7. Conclusions

Publishing datasets that contain demographics and diagnosis codes (RT-datasets) is important, in the context of several medical analysis tasks. To preserve the privacy and utility of RT-datasets, we proposed an approach that enforces \((k, k^m)\)-anonymity, while satisfying intended analysis requirements with minimal information loss. In particular, we introduced the concept of utility constraints for RT-datasets, to limit the amount of data generalization, and developed an algorithm that constructs \((k, k^m)\)-anonymous clusters of records, using generalization and suppression. Experiments using a dataset containing over 200,000 electronic health records showed that our algorithm is effective at preserving data utility and also efficient.

Appendix A. \((k, k^m)\)-anonymization

In this appendix, we devise a method for \((k, k^m)\)-anonymizing a cluster of records in a way that satisfies the utility constraint set and incurs minimal information loss according to the \( UL \) measure. Our method is called \( GsnDoc \) and is based on the CBA algorithm [45], discussed in Section 2.2. \( GsnDoc \) differs from CBA in that it enforces \( k^m \)-anonymity. The pseudocode of our method is illustrated in Algorithm \( GsnDoc \). Our method takes as input a cluster of records \( C \), a utility constraint set \( U \), and the anonymization parameters \( k \) and \( m \), and it outputs a \((k, k^m)\)-anonymous cluster \( C' \) corresponding to \( C \) and the number of diagnosis codes \( s \) that have been suppressed from \( C \).
Algorithm: GenDiag
Input: A cluster of records \( C \), a utility constraint set \( U \), and the anonymization parameters \( k \) and \( m \).
Output: A \((k, k^m)\)-anonymous cluster \( C' \) corresponding to \( C \) and the number of diagnosis codes \( s \) that have been suppressed from \( C \).

1. Initialize \( C' = C \) and \( s = 0 \).
2. Let \( Q \) be the set that contains sets of diagnosis codes, each of which contains up to \( m \) codes and appears in fewer than \( k \) records of \( C' \).
3. While \( Q \) is not empty do
   
   4. Let \( p \) be the element of \( Q \) that appears in the largest number of records.
   
   5. While \( p \) appears in fewer than \( k \) records of \( C' \) do
       
       6. Find codes \( u \) and \( u' \), such that (a) \( u \) is contained in \( p \), (b) \( u \) and \( u' \) are contained in the same utility constraint of \( U \), and (c) \( UL(u) \) is minimum, where \( UL \) is the generalized diagnosis code containing the set of diagnosis codes that appear in \( u \) or \( u' \).
       
       7. If \( u \) and \( u' \) are found then
           
           8. Replace \( u \) and \( u' \) by \( \bar{u} \) in \( p \), \( Q \) and in all records of \( C' \).
           
       9. Else while \( p \) appears in fewer than \( k \) records of \( C' \) do
           
           10. Let \( e \) be the element of \( p \) that appears in the minimum number of records of \( C \).
                
               11. Increase \( s \) by the number of diagnosis codes that appear in \( e \).
               
               12. Suppress \( e \) from \( p \), \( Q \) and all records of \( C' \).
                
           13. \( \vdots \)
           
       14. Remove \( p \) from \( Q \).

4. Return \( C' \) and \( s \).

In Step 1, GenDiag initializes the output variables \( C' \) and \( s \), and in Step 2, it finds all sets of up to \( m \) diagnosis codes that appear in fewer than \( k \) records of the cluster and stores them in set \( Q \). Next, it iterates over these sets, in decreasing order of frequency (number of records of the cluster in which these sets appear) and performs generalization and/or suppression (Steps 3–14).

Specifically, for a set of diagnosis codes \( p \) with element \( u \), a generalized diagnosis code \( \bar{u} \) is created (Steps 6–8). The generalized diagnosis code \( \bar{u} \) contains all the diagnosis codes of \( u \) as well as additional codes from the same utility constraint (encoded by code \( u' \) in Step 6), so that the utility constraint set \( U \) is satisfied. Additionally, GenDiag chooses \( u \) and \( u' \) so that their generalization \( \bar{u} \) incurs minimal information loss (Step 6). If \( u \) and \( u' \) are found then the generalization is performed (Steps 7 and 8). Otherwise, GenDiag iteratively suppresses, from the element \( p \), the diagnosis code that appears in the minimum number of records in the cluster, and it keeps, in variable \( s \), the summation of suppressed diagnosis codes (Steps 9–12).

Generalization and suppression are performed until the set of codes \( p \) appears in at least \( k \) records of the cluster and, after that, \( p \) is not considered again (Step 14). When the loop of Steps 3–14 terminates, cluster \( C \) is \((k, k^m)\)-anonymous. Thus, GenDiag outputs cluster \( C' \) and the total number of suppressed diagnosis codes \( s \) (Step 15).

Example 6. Let us apply GenDiag to the set of records \{0, 1, 2, 3\} of Fig. 3a using the utility constraint set of Table 4 and parameters \( k = 2 \) and \( m = 2 \) (the input set corresponds to the union of clusters \( C_1 = \{0, 1\} \) and \( C_2 = \{2, 3\} \)) formed in the example of Section 4.2.

Initially, the set of diagnosis codes \{(493.2, 053.20, 053.71)\}, which is contained in fewer than two records, is added into \( Q \) (Step 2). Next, the element \( p = (493.2, 053.20) \) is retrieved, the pair \( 053.20, 053.71 \), and the generalized diagnosis code \((053.20, 053.71)\) is created (Steps 4–6). To verify, note that \( 053.71 \) is contained in the same utility constraint with \( 053.20 \) and that the \( UL \) of \((053.20, 053.71)\) is minimum. Note also that \( 493.2 \) (i.e., the other code in \( p \)) cannot be generalized with any other diagnosis code, as this would violate the utility constraint set of Table 4.

Following that, GenDiag replaces all occurrences of the code \( 053.20 \) in the input set with \((053.20, 053.71)\), and \( p \) becomes \((493.2, (053.20, 053.71)) \) (Step 8). Since \( p \) now appears in three records, it is removed from \( Q \) (Steps 14), and GenDiag returns the anonymized output set (illustrated by the first 4 rows of the dataset in Fig. 3).

Appendix B. Complexity of GenDiag

Let \( C \) be the input cluster, \( \beta \) be the maximum number of diagnosis codes contained in any record of the cluster \( C \), and \( \gamma \) be the total number of distinct diagnosis codes in the records of the cluster \( C \). Let also \( Q \) be the set that is comprised of sets of diagnosis codes, each of which contains up to \( m \) codes and appears in fewer than \( k \) records of \( C \). Last, let \( p \) be the element of \( Q \) that contains the larger number of diagnosis codes.

The first two steps of GenDiag need time \( O\left(|C| \cdot \sum_{i=1}^{m} \left(\frac{\beta}{i}\right) + |Q| \cdot \log(|Q|)\right) \) time. The first term, in square brackets, is the time to construct \( Q \) (Step 2), and the second term is the time to sort \( Q \). Sorting is required by the next steps of GenDiag, to access the elements of \( Q \) in decreasing frequency order. The while loop of Steps 3–14 is performed \( O(|Q|) \) times, and each iteration of Steps 5–12 needs \( O(|p|^2 \cdot \gamma + |p| \cdot |C|) \) time. Thus, the while loop of Steps 3–14 takes \( O(|Q| \cdot (|p|^2 \cdot \gamma + |p| \cdot |C|)) \) time.

Therefore, the time complexity of GenDiag is:

\[ O\left(|C| \cdot \sum_{i=1}^{m} \left(\frac{\beta}{i}\right) + |Q| \cdot \log(|Q|) + |p|^2 \cdot \gamma + |p| \cdot |C|\right) \]

or equivalently:

\[ O\left(|C| \cdot \sum_{i=1}^{m} \left(\frac{\beta}{i}\right) + |Q| \cdot \log(|Q|) + m^2 \cdot \gamma + m \cdot |C|\right) \]

since \( |p| \) is at most \( m \).

Appendix C. Time complexity of ArtDC

To compute the time complexity of ArtDC, we compute the complexity of each of its phases. In the following, we denote the size of a given a set \( S \) with \( |S| \).
Record grouping requires a single pass of the RT-dataset. Thus, it can be performed in $O(D)$ time.

Cluster formation requires $O(\sum_{G \in G} |G|^2)$ time, where $G$ is the set of clusters formed in Steps 1 and 2. For each group $G \in G$, ARTUC compares a random record with all others in the set (Step 8) which can be performed in $O(|G|^2)$ time. Note that the generalization with Definition 2 is performed in an incrementally fashion. Thus, it can be performed in constant time. Also, the final steps 15 and 16 need $O(|C|)$ time and do not affect the overall complexity.

Cluster merging Step 17 can be implemented with a multidimensional sorting in $O(|\cdot| \cdot \log(|\cdot|))$ time, where $i$ is the number of dimensions, i.e., the number of demographic attributes. The repeat/until loop of Steps 18–22 is executed at most $O(|C|^2)$ times. The time needed to execute Steps 19–22 once is $O(\mu)$, where $O(\mu)$ is the cost of executing our adaptation of the CBA algorithm with input the two larger clusters of set $C$ (see Appendix A for more details). In summary, the repeat/until loop can be done in $O(|C|^2 \cdot \mu)$ time. Thus in total, the cluster merging step can be performed in $O(|C|^2 \cdot \mu)$ time.

(k, $k$)-anonymization requires $O(\sum_{G \in G}(cba(C)))$ time.

Therefore, the cost time consuming phases of ARTUC are cluster formation and merging, and the time complexity of ARTUC is $O(\sum_{G \in G}(G^2) + |C|^2 \cdot \mu)$.

References
